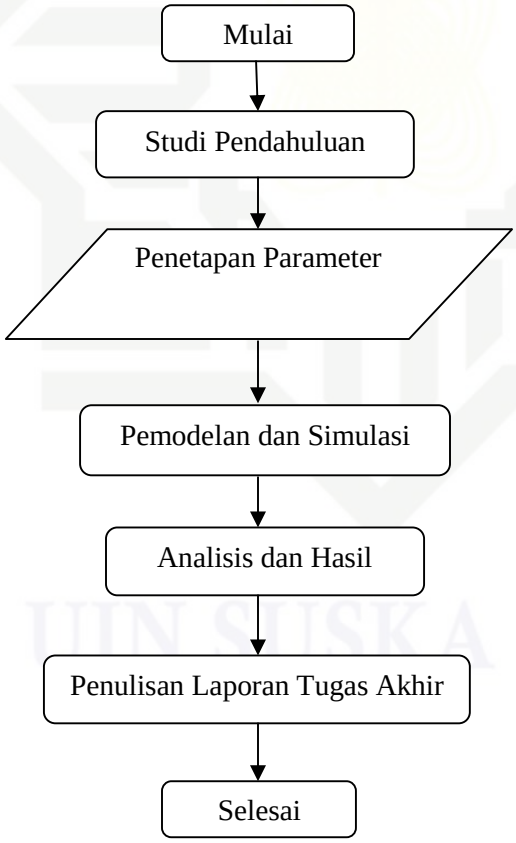
BAB III
METODOLOGI PENELITIAN

Metodologi penelitian merupakan suatu cara berpikir yang di mulai dari menentukan suatu permasalahan, pengumpulan data baik dari buku-buku panduan maupun studi lapangan, melakukan penelitian berdasarkan data yang ada sampai dengan penarikan kesimpulan dari permasalahan yang diteliti. Dalam metode penelitian direncanakan cara atau prosedur beserta tahapan-tahapan yang jelas dan disusun secara sistematis dalam proses penelitian. Tiap tahapan merupakan bagian yang menentukan tahap berikutnya sehingga harus dilalui dengan cermat.

3.1 Langkah Penelitian

Langkah-langkah yang akan dilakukan pada penelitian dapat dilihat pada gambar di bawah ini :



Gambar 3.1 *Flowchart* Penelitian

Penjelasan Gambar 3.1 *Flowchart* Penelitian :

3.1.1 Studi Pendahuluan

Adapun tujuan dari studi pendahuluan yaitu untuk memperoleh bahan mengenai sistem SC-FDMA yang akan diteliti dalam menentukan sebuah penelitian. Sehingga diharapkan dapat memperoleh informasi mengenai permasalahan sistem SC-FDMA yang akan diangkat dalam penelitian yang terkait dengan permasalahan tersebut. Dalam Tugas Akhir ini studi pendahuluan yang akan dilakukan yaitu mencari referensi pada penelitian sebelumnya tentang SC-FDMA, guna sebagai bahan acuan pada Tugas Akhir ini.

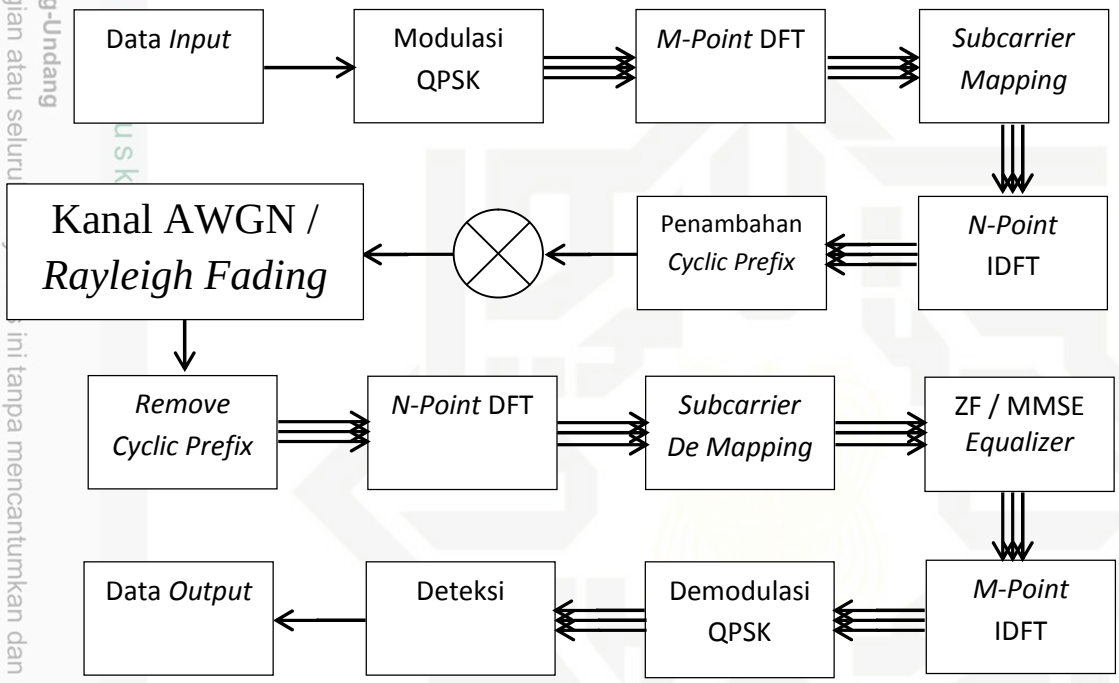
3.1.2 Penetapan Parameter

Menentukan parameter yang akan diukur pada Tugas Akhir ini adalah untuk mengetahui kinerja dari sistem SC-FDMA. Pada sistem SC-FDMA digunakan beberapa asumsi dan parameter yaitu :

1) Input data biner yang dibangkitkan sebanyak 1000 bit
2) Modulasi yang digunakan adalah QPSK
3) Jumlah *M-Point* IDFT dan *N-Point* DFT pada sistem SC-FDMA adalah 1024 titik
4) Menggunakan *guard interval* dengan *cyclic prefix* ¼ dari jumlah FFT
5) Pada kondisi sistem SC-FDMA dengan ZF *equalizer* dilewatkan di kanal AWGN
6) Pada kondisi sistem SC-FDMA dengan ZF *equalizer* dilewatkan di kanal *Rayleigh Fading*
7) Pada kondisi sistem SC-FDMA dengan MMSE *equalizer* dilewatkan di kanal AWGN
8) Pada kondisi sistem SC-FDMA dengan MMSE *equalizer* dilewatkan di kanal *Rayleigh Fading*
9) Dalam sistem SC-FDMA ini, *transmitter* dan *receiver* diasumsikan berada dalam keadaan tetap (*fixed*)
10) Input berupa SNR yang akan divariasikan dari 0 dB hingga 30 dB
11) Perhitungan BER menggunakan metode *Monte Carlo*

3.1.3 Pemodelan dan Simulasi

Dalam sistem SC-FDMA terdapat tiga komponen utama yaitu blok pengirim, kanal transmisi dan blok penerima. Secara umum sistem SC-FDMA dan OFDMA memiliki kesamaan, akan tetapi yang membedakan adalah adanya penambahan blok DFT di *transmitter* dan blok IDFT pada *receiver* sistem SC-FDMA. Simulasi pada penelitian ini menggunakan *software* MATLAB 2013a.



Gambar 3.2 Perancangan Sistem SC-FDMA menggunakan ZF *Equalizer* dan MMSE *Equalizer*

3.1.4 Analisis dan Hasil

Analisis dilakukan dari hasil simulasi antara ZF *equalizer* dan MMSE *equalizer* menggunakan modulasi QPSK pada MATLAB 2013a. Perbandingan diperoleh dari kedua teknik *equalizer* ini sehingga dapat diketahui dari kedua *equalizer* ini yang mempunyai kinerja yang lebih baik dan dapat digunakan sebagai referensi apabila *equalizer* tersebut nantinya digunakan pada jaringan *uplink* LTE.

3.1.5 Penulisan Laporan Tugas Akhir

Penulisan laporan Tugas Akhir dari hasil yang sudah diperoleh yaitu dari pengumpulan data-data yang dibutuhkan, hasil analisis, hasil simulasi, dan hasil perbandingannya.

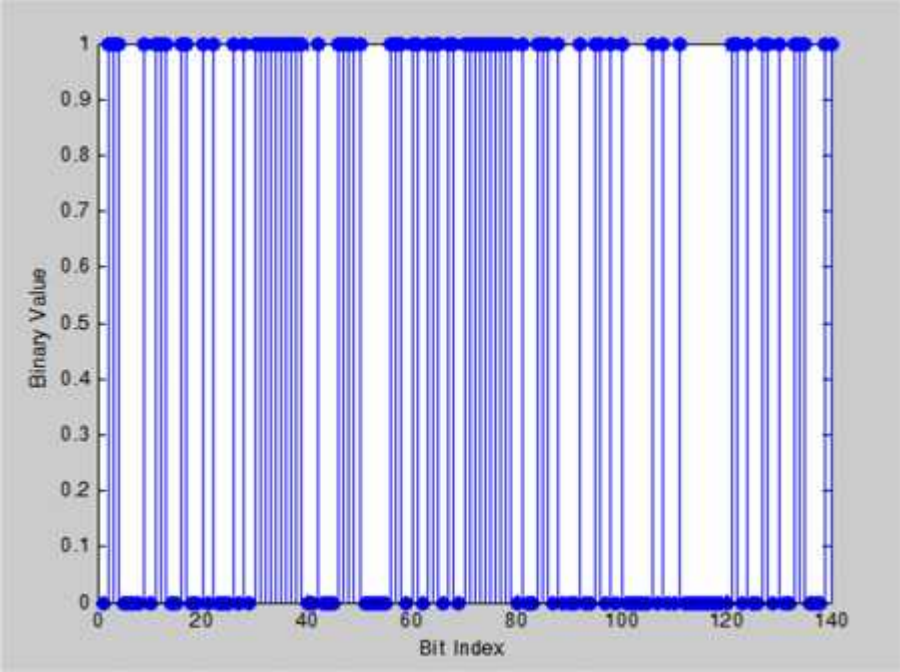
3.2 Pembangkitan Bit Informasi

3.2.1 Pembangkitan Data

Pada simulasi ini, akan dilakukan menggunakan program MATLAB 2013a sinyal informasi dapat dibangkitkan setelah jumlah bit informasi telah diketahui. Jumlah bit informasi yang akan dibangkitkan diasumsikan sebanyak 1000 bit.

Pada sistem SC-FDMA ini dibangkitkan bit informasi dengan fungsi pada MATLAB :

```
%%Parameter
FFTsize = 1024; % The size of the transmitter IFFT and the
receiver FFT.
inputBlockSize = 32; % Input data block size.
CPsize = 1/4*FFTsize; % CP length.
subband = 0; % Set the subband location.
SNR = [0:2:30]; % Simulated SNR range is from 0 dB to 30 dB.
numRun = 10^3; % The number of simulation iterations is 10^3.
```



Gambar 3.3 Bit yang dikirimkan

Dalam MATLAB, perintah untuk mensimulasikan pada gambar 3.3 adalah sebagai berikut :

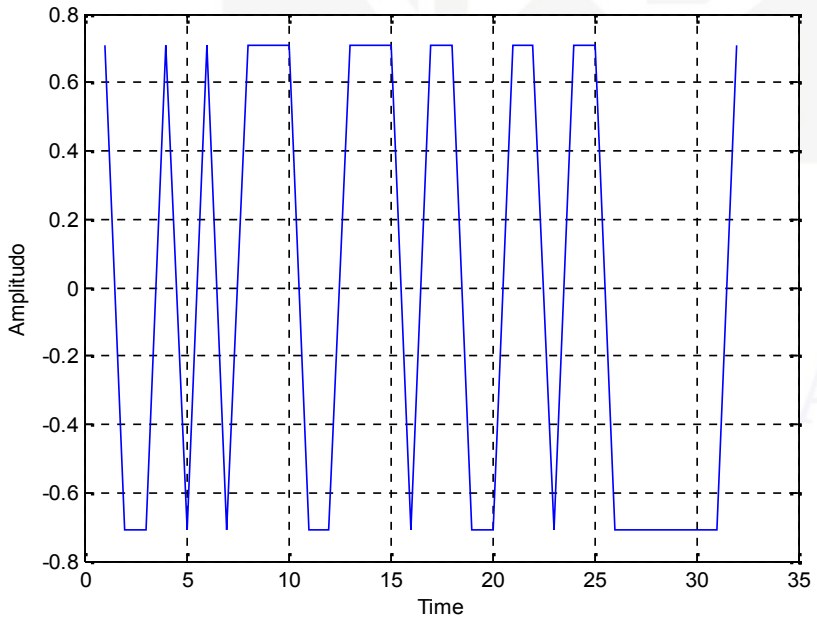
```
s=randint(numRun,1);
stem(s(1:140),'filled');
xlabel('Bit Index');
ylabel('Binary Value');
```

3.2.2 Modulasi Data

Sebelum bit informasi akan dikirim, bit informasi akan dimodulasi terlebih dahulu. Dalam Tugas akhir ini modulasi digital yang digunakan pada bit informasi adalah modulasi QPSK, data dikirim melewati kanal AWGN dan *Rayleigh Fading*. Pada modulasi tersebut, 2 bit informasi dipetakan menjadi 1 simbol data sehingga bit informasi yang berjumlah 1000 bit akan berjumlah 500 simbol data setelah dimodulasi menggunakan modulasi QPSK.

Untuk membangkitkan modulasi QPSK digunakan fungsi pada MATLAB :

```
% Modulasi QPSK.
tmp = round(rand(2,inputBlockSize));
tmp = tmp*2 - 1;
inputSymbols = (tmp(1,:) + i*tmp(2,:))/sqrt(2);
```



Gambar 3.4 Sinyal Output Proses Modulasi

Dalam MATLAB untuk membangkitkan sinyal pada gambar 3.4 dapat dilakukan dengan perintah :

```
T=1:32;
Z= inputSymbols(1:32);
plot(T,Z,'blue');
xlabel('Time');
ylabel('Amplitudo');
grid on;
```

3.2.3 *M-Point* DFT

DFT berfungsi untuk mengubah sinyal dalam domain waktu menjadi domain frekuensi. Selain itu juga berfungsi untuk membuat frekuensi *multiplexing* atau *multiple access*. Walaupun menggunakan *single-carrier*, setiap user di *multiplexing* dengan frekuensi yang berbeda-beda. Blok DFT ini yang membedakan antara sistem SC-FDMA dan OFDMA.

Untuk membangkitkan *M-point* DFT digunakan fungsi MATLAB :

```
% DFT-precoding.
inputSymbols_freq = fft(inputSymbols);
```

3.2.4 *Subcarrier Mapping*

Subcarrier mapping ada 2 macam, yaitu *distributed mode* dan *localized mode*. Pada *distributed mode*, *subcarrier-subcarrier* yang dialokasikan untuk setiap user ditempatkan tersebar di seluruh frekuensi *bandwidth*, dan tercampur dengan *subcarrier-subcarrier* dari user lainnya. Sedangkan pada *localized mode, subcarrier-subcarrier* yang dialokasikan untuk setiap user ditempatkan secara berkelompok dan berurutan sesuai dengan urutan *user* sehingga tidak tercampur dengan *subcarrier-subcarrier* lainnya. Dalam Tugas Akhir ini, *subcarrier mapping* yang digunakan adalah *localized mode*, yaitu menggabungkan semua *subcarrier* dari tiap-tiap *user* secara berurutan.

Dalam simulasi *subcarrier mapping*, digunakan fungsi pada MATLAB :

```
% Initialize the output subcarriers.
inputSamples_lfdma = zeros(1,numSymbols);
% Subcarrier mapping (Localized mode).
```

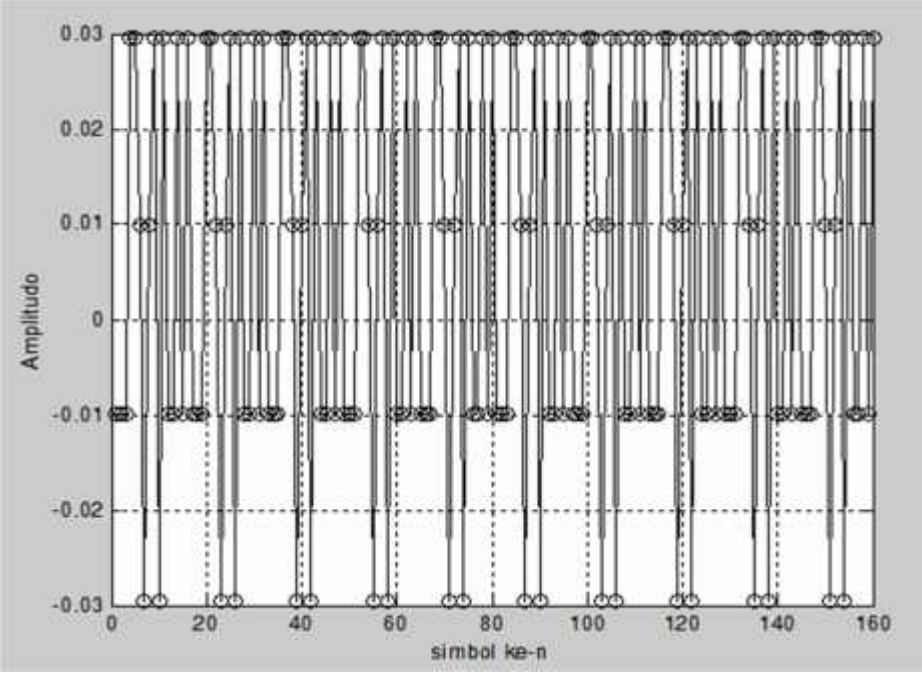
```
inputSamples_lfdma([1:inputBlockSize]+inputBlockSize*subband) =
inputSymbols_freq;
```

3.2.5 *N-Point* IDFT

Proses selanjutnya setelah data melalui proses *subcarrier mapping* adalah proses *N-point* IDFT. Fungsi proses IDFT disini adalah sebagai modulator SC-FDMA dengan cara mengalikan setiap *subcarrier user* dengan frekuensi yang berbeda-beda kemudian seluruh *subcarrier* tersebut dijumlahkan. Sehingga didapat *subcarrier* yang saling *orthogonal* dalam domain frekuensi tanpa saling berinterferensi.

Dalam simulasi *N-point* IDFT digunakan fungsi MATLAB :

```
% Proses IFFT.
inputSamples_lfdma = ifft(inputSamples_lfdma);
```



Gambar 3.5 Sinyal Output Proses *N-Point* IDFT

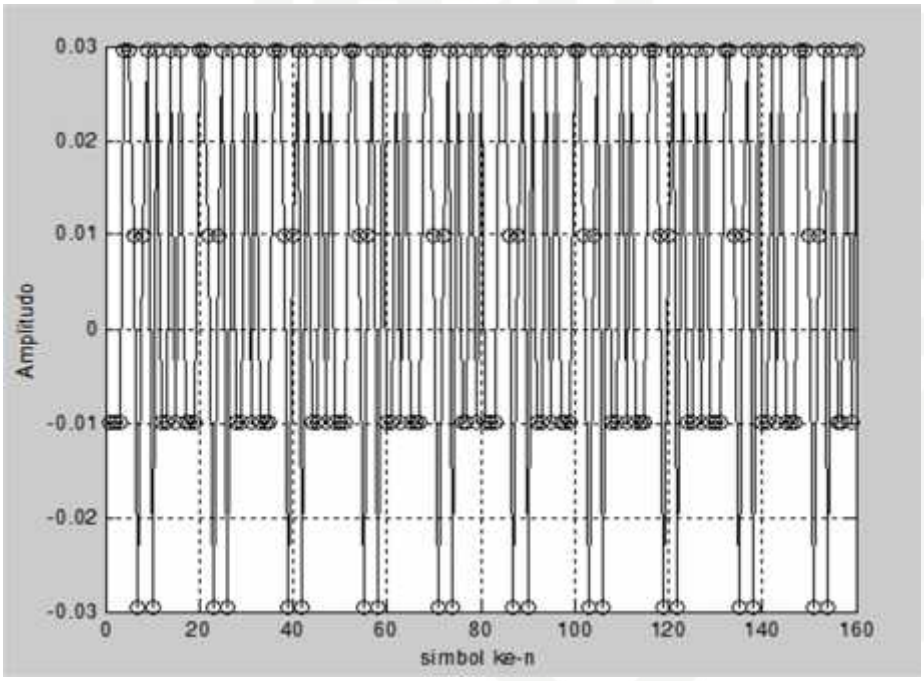
3.2.6 Penambahan *Cyclic Prefix*

Pada implementasi simulasi ini penambahan *guard interval* dengan *cyclic prefix* dilakukan setelah proses *N-point* IDFT. Pada simulasi ini, proses *insert prefix* dilakukan dengan cara mengambil ¼ jumlah titik *M-point*.

Jumlah *M-point* pada inisialisasi program MATLAB sebanyak 1024 titik maka jumlah *cyclic prefix* yang ditambahkan 256. Proses penambahan *guard interval* dengan *cyclic prefix* adalah keempat N paling bawah dari periode IFFT kemudian keempat N tersebut digabungkan dengan hasil IFFT dan diletakkan pada bagian awal sebelum N pertama dari IFFT, sehingga periode total adalah periode simbol dari IFFT ditambah periode *guard interval* dengan *cyclic prefix.*

Untuk membangkitkan fungsi penambahan *cyclic prefix* digunakan fungsi MATLAB:

```
% Add CP.
TxSamples_lfdma = [inputSamples_lfdma(numSymbols -
CPsize+1:numSymbols) inputSamples_lfdma];
```



Gambar 3.6 Sinyal Output Setelah Penambahan *Cyclic Prefix*

3.2.7 Kanal AWGN

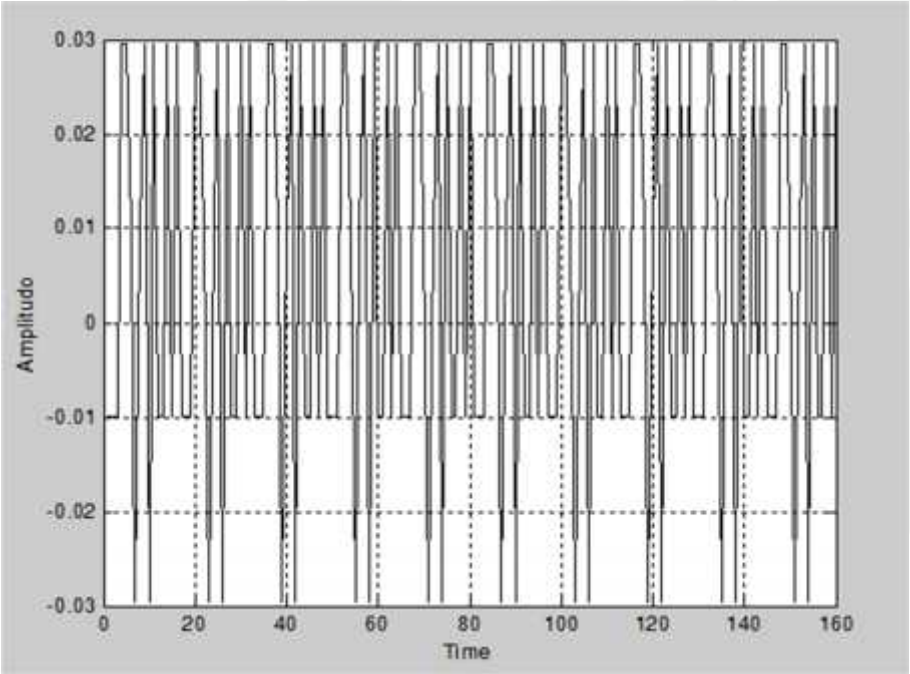
Dalam simulasi sistem SC-FDMA pada kondisi kanal pertama, kanal yang digunakan adalah kanal AWGN. Pada pemodelan ini, *noise* AWGN dibangkitkan secara acak dengan menggunakan fungsi AWGN dengan menggunakan nilai SNR tertentu yang bervariasi. *Noise* AWGN bersifat menambahkan amplitude sinyal melalui persamaan berikut:

$$r(t) = s(t) + n(t)$$

Dengan *r(t)* adalah sinyal yang diterima setelah terkena *noise* AWGN, *s(t)* adalah sinyal yang ditransmisikan sebelum terkena *noise* AWGN. Sedangkan *n(t)* adalah *noise* AWGN.

Untuk membangkitkan kanal AWGN digunakan fungsi pada MATLAB :

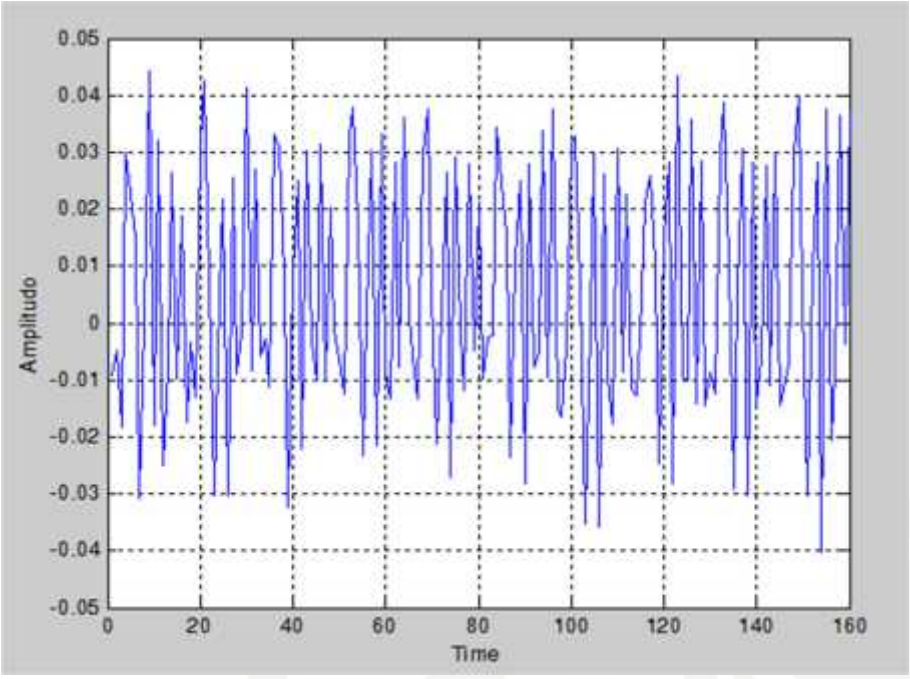
```
% Propagate through multi-path channel.
RxSamples_lfdma = filter(channel, 1, TxSamples_lfdma);
% Generate AWGN with appropriate noise power.
tmp = randn(2, numSymbols+CPsize);
complexNoise = (tmp(1,:) + i*tmp(2,:))/sqrt(2);
noisePower = 10^(-SNR(n)/10);
% Add AWGN to the transmitted signal.
RxSamples_lfdma = RxSamples_lfdma +
sqrt(noisePower/Q)*complexNoise;
```



Gambar 3.7 Sinyal SC-FDMA Sebelum Pengaruh *Noise* AWGN

Dalam MATLAB untuk membangkitkan sinyal pada gambar 3.7 dapat dilakukan dengan perintah :

```
T=1:160; Z= RxSamples_lfdma (1:160);
plot(T,Z,'black'); xlabel('Time');
ylabel('Amplitudo'); grid on;
```



Gambar 3.8 Sinyal SC-FDMA Setelah Pengaruh *Noise* AWGN

Dalam MATLAB, perintah untuk mensimulasikan kondisi tersebut adalah :

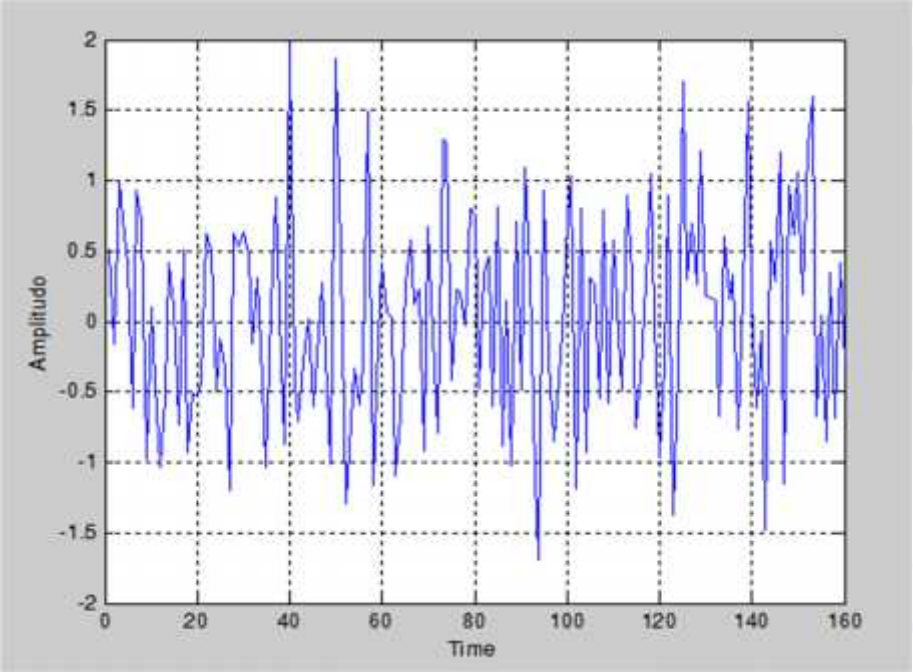
```
T=1:160;
S= RxSamples_lfdma(1:160);
plot(T,S,'b');
xlabel('Time');
ylabel('Amplitudo');
grid on;
```

3.2.8 Kanal *Rayleigh Fading*

Dalam simulasi ini, sistem SC-FDMA pada kondisi kanal yang kedua, kanal yang digunakan adalah kanal *Rayleigh Fading*. Kanal *Fading* di pengaruhi oleh dua karakteristik *fading* yaitu *fading* dalam skala luas dan skala kecil, *Fading* skala besar diasumsikan dalam proses yang lambat dan biasanya dimodelkan secara statisitik dengan log normal. *Fading* dalam skala kecil juga disebut sebagai *Rayleigh Fading*, karena sebagian besar selubung sinyal yang diterima adalah lintasan pantul dan dapat digambarkan dengan fungsi kerapatan probabilitas (PDF) *Rayleigh*. *Fading* bisa disebut *Rayleigh*, bila lintasan propagasi *Line Of Sight* (LOS) tidak dominan, sedangkan *Rician*, lintasan propagasi LOS-nya lebih dominan.

Untuk membangkitkan kanal *Rayleigh Fading* digunakan fungsi MATLAB :

```
% Generate AWGN with appropriate noise power.
tmp = randn(2, numSymbols+CPsize);
complexNoise = (tmp(1,:) + i*tmp(2,:))/sqrt(2);
noisePower = 10^(-SNR(n)/10);
% Impuls respon kanal rayleigh fading.
h =
1/sqrt(2)*[randn(1,length(RxSamples_lfdma))+j*randn(1,length(Rx
Samples_lfdma))];
RxSamples = ((RxSamples_lfdma.*h) +
sqrt(noisePower/Q)*complexNoise);
% Receiver.
d = RxSamples./h;
% Add AWGN to the transmitted signal.
RxSamples_lfdma = d + sqrt(noisePower/Q)*complexNoise;
```



Gambar 3.9 Respon *Impuls* Kanal *Rayleigh Fading*

Untuk membangkitkan gambar 3.9 digunakan fungsi pada MATLAB :

```
T=1:160;
hh= h(1:160);
plot(T,hh,'b');
xlabel('Time');
ylabel('Amplitudo');
grid on;
```

3.2.9 *Remove Cyclic Prefix*

Proses penghilangan *guard interval* dengan *cyclic* adalah untuk mengembalikan jumlah titik IFFT agar sesuai dengan jumlah titik pemancar hasil dari IFFT. Karena pada pemancar ditambah dengan proses *guard interval* dengan *cyclic prefix*, maka jumlah titik menjadi lebih besar dari data informasi yang dikirim. Untuk mengembalikannya seperti semula maka jika jumlah titik semula adalah 1024.

Untuk *remove cyclic prefix* digunakan fungsi pada MATLAB :

```
% Remove CP.
RxSamples_lfdma = RxSamples_lfdma(CPsize+1:numSymbols+CPsize);
```

3.2.10 Proses *N-Point* DFT

Pada proses *N-Point* DFT kebalikan dari *N-Point* IDFT yang mengalikan setiap *subcarrier* dengan frekuensi yang berbeda. Pada proses ini mengembalikan *subcarrier* yang telah dikalikan dengan frekuensi yang berbeda-beda yang berguna mendapatkan *subcarrier* yang terdapat dipemancar.

Untuk membangkitkan proses *N-Point* DFT digunakan fungi pada MATLAB :

```
% Proses FFT.
Y_lfdma = fft(RxSamples_lfdma, FFTsize);
```

3.2.11 *Subcarrier Demapping*

Dalam proses ini merupakan proses kebalikan dari proses *subcarrier mapping* pada *transmitter*. Walaupun menerima keseluruhan spektrum dari BTS, setiap *user* kemudian menyeleksi *subcarrier* yang diperuntukan baginya saja. Sebelum melalui proses *subcarrier*

demapping, subcarrier selection terlebih dahulu melalui proses FFT yang membuatnya berada dalam domain frekuensi. Data berupa data paralel yang berukuran 1024 x 1000 data. Dari ukuran tersebut selanjutnya akan diseleksi sebanyak *N-Point* DFT setiap *user*. Alokasi *subcarrier* yang diproses pada *transmitter* adalah 1024 *N-DFT*.

Fungsi yang digunakan pada MATLAB untuk membangkitkan *subcarrier demapping* adalah :

```
% Subcarrier de-mapping.
Y_lfdma = Y_lfdma([1:inputBlockSize]+inputBlockSize*subband);
```

3.2.12 *Equalizer*

Fungsi dari teknik *equalizer* adalah untuk memperbaiki data yang rusak akibat distorsi kanal. Pada *receiver* akan di lakukan teknik *equalizer* pada simbol-simbol yang diterima sehingga dapat mengurangi ISI yang dapat menyebabkan kesalahan pembacaan bit pada penerima. Teknik *equalizer* yang akan digunakan adalah ZF *equalizer* dan MMSE *equalizer*.

3.2.12.1 ZF *Equalizer*

ZF *equalizer* merupakan *equalizer* paling sederhana untuk meminimalkan distorsi. Kinerja ZF *equalizer* dalam menghilangkan ISI pada sistem SC-FDMA dapat dicapai dengan sinyal yang diterima dibagi dengan respons impuls kanal.

Untuk proses penambahan ZF *equalizer* digunakan fungsi pada MATLAB :

```
% Find the channel response for the localized mode subcarriers.
H_eff1 =
H_channel1([1:inputBlockSize1]+inputBlockSize1*subband1);
% Perform channel equalization in the frequency domain.
equalizerType == 'ZERO'
Y_lfdma1 = Y_lfdma1./H_eff1;
```

3.2.12.2 MMSE Equalizer

Kinerja MMSE *equalizer* dalam menghilangkan ISI pada sistem SC-FDMA dapat di lakukan dengan cara mengalikan nilai sinyal yang diterima dengan nilai MMSE *equalizer*.

Untuk proses penambahan MMSE *equalizer* digunakan fungsi pada MATLAB :

```
% Find the channel response for the localized mode subcarriers.
H_eff2 =
H_channel2([1:inputBlockSize2]+inputBlockSize2*subband2);
% Perform channel equalization in the frequency domain.
equalizerType == 'MMSE'
C2 = conj(H_eff2)./(conj(H_eff2).*H_eff2 + 10^(-SNR2(n2)/10));
Y_lfdma2 = Y_lfdma2.*C2;
```

3.2.13 Proses *M-Point* IDFT

Proses selanjutnya adalah *M-point* IDFT. Proses ini hanya terdapat pada sistem SC-FDMA. IDFT berfungsi untuk mengubah sinyal domain frekuensi menjadi sinyal domain waktu. Selain itu juga sebagai frekuensi *demultiplexing*, setiap *user* di-*demultiplexing*-kan dengan frekuensi yang berbeda-beda. Sehingga pemrosesan sinyal selanjutnya dilakukan di domain waktu.

Setelah sinyal telah mengalami proses IDFT maka selanjutnya sinyal akan di *desampling*. Artinya sinyal direpresentasikan kembali seperti bentuk semula.

Untuk membangkitkan proses *M-Point* IDFT digunakan fungsi pada MATLAB :

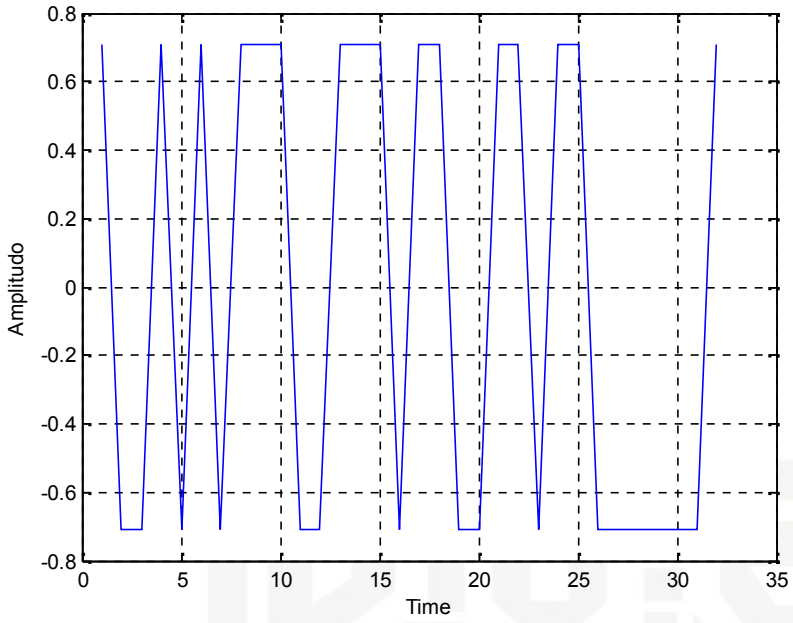
```
% Proses IFFT.
EstSymbols_lfdma = ifft(Y_lfdma);
```

3.2.14 Demodulasi

Pada proses ini sinyal akan di demodulasi ke bentuk semula sesuai dengan data yang dikirimkan tanpa ada *carrier* yang telah ditambahkan pada sisi modulasi.

Untuk membangkitkan proses demodulasi digunakan fungsi pada MATLAB :

```
% Proses demodulasi.
EstSymbols_lfdma = sign(real(EstSymbols_lfdma)) +
i*sign(imag(EstSymbols_lfdma));
EstSymbols_lfdma = EstSymbols_lfdma/sqrt(2);
```



Gambar 3.10 Sinyal Output Proses Demodulasi

Dalam MATLAB untuk membangkitkan sinyal pada gambar 3.10 dapat dilakukan dengan perintah :

```
T=1:32;
Z= EstSymbols_lfdma(1:32);
plot(T,Z,'blue');
xlabel('Time');
ylabel('Amplitudo');
grid on;
```

3.2.15 Perhitungan *Bit Error Rate* (BER)

BER dihitung dengan menggunakan metode *monte carlo*, yaitu dengan membandingkan antara deretan bit pada pengirim dengan deretan bit yang dideteksi pada sisi penerima, kemudian jumlah bit yang salah dibagi dengan jumlah bit yang dibangkitkan.

Untuk menghitung BER pada MATLAB digunakan perintah sebagai berikut :

```
% Find and count errors.
I_lfdma = find((inputSymbols-EstSymbols_lfdma) == 0);
```

```
 errCount_lfdma = errCount_lfdma + (inputBlockSize-
length(I_lfdma));
end
% Calculate the Bit Error Rate (BER).
BER_lfdma(n,:) = errCount_lfdma / (inputBlockSize*numRun);
end
figure;
semilogy(SNR,BER_lfdma,'bd-','Linewidth',2);
ylabel('Bit Error Rate');
xlabel('Eb/No, dB');
grid on;
```

3.2.16 Perhitungan *Peak-to-Average Power Ratio* (PAPR)

Perhitungan PAPR pada sistem SC-FDMA dilakukan pada sisi pemancar / *transmitter* dengan cara membandingkan nilai maksimum daya puncak sinyal dengan daya rata-rata sinyal. Dalam Tugas Akhir ini, perhitungan nilai PAPR tidak dilakukan untuk setiap simbol SC-FDMA, melainkan dilakukan untuk semua sinyal pada *transmitter* SC-FDMA yang terdiri dari 1024 simbol.

Untuk mengetahui nilai puncak maksimum sinyal digunakan perintah `max(abs(RxSamples_lfdma).^2)`pada MATLAB dan untuk mengetahui nilai daya rata-rata sinyal digunakan perintah `mean(abs(RxSamples_lfdma).^2)` pada MATLAB. Sedangkan untuk menghitung nilai PAPR yaitu dengan membandingkan daya puncak maksimum dengan daya puncak rata-rata sinyal.